Chapter 2

Optimization

Gradients, convexity, and ALS

max planck institut
informatik

Contents

- Background
- Gradient descent
- Stochastic gradient descent
- Newton's method
- Alternating least squares
- KKT conditions

Motivation

- We can solve basic least-squares linear systems using SVD
- But what if we have
  - missing values in the data
  - extra constraints for feasible solutions
  - more complex optimization problems (e.g. regularizers)
  - etc

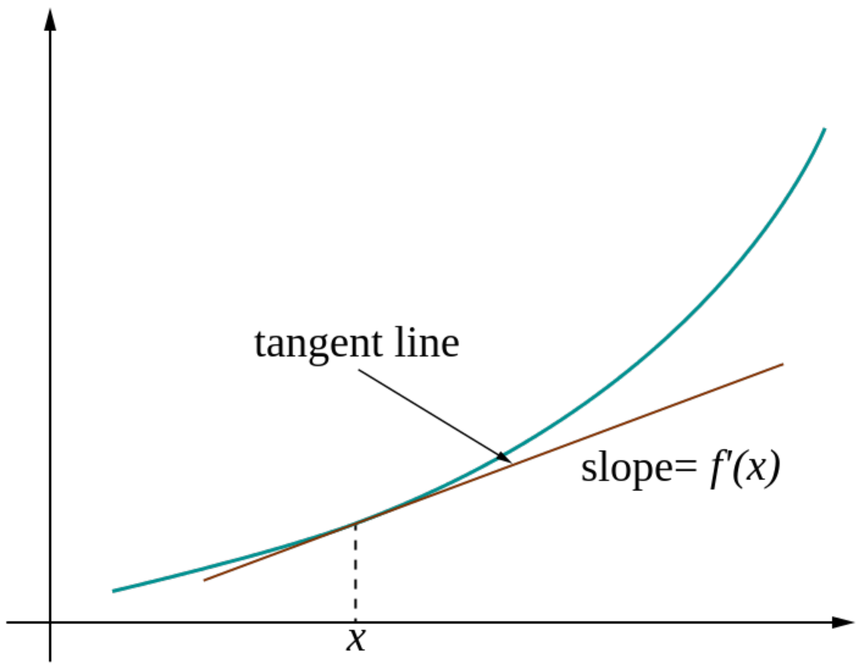
Gradients, Hessians, and convexity

Derivatives and local optima

- The **derivative** of a function *f*: ℝ → ℝ, denoted *f'*, explains its *rate of change*

$$f'(a) = \lim_{h \to 0+} \frac{f(a+h) - f(a)}{h}$$

  - If it exists
- The second derivative *f''* is the change of rate of change



Derivatives and local optima

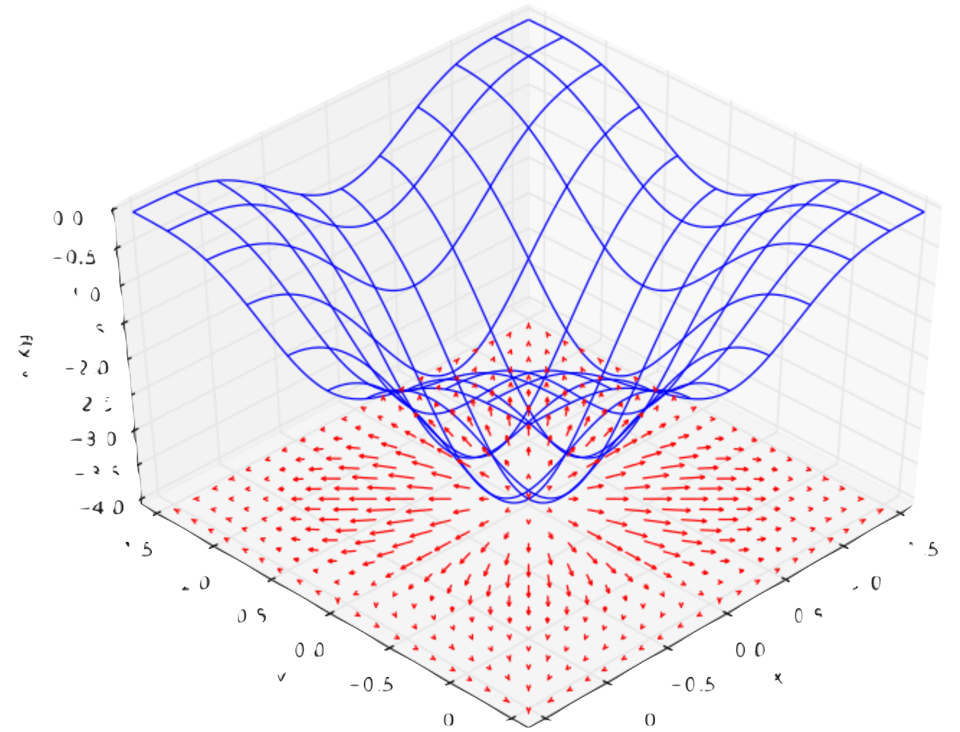
- A **stationary point** of differentiable f is x s.t. $f'(x) = 0$
  - f achieves its extremes in stationary points or in points where derivative doesn't exist, or at infinities (Fermat's theorem)
- Whether this is (local) maximum or minimum can be seen from the second derivative (if it exists)

Partial derivative

- If f is multivariate (e.g. $f: \mathbb{R}^3 \to \mathbb{R}$), we can consider it as a family of functions
  - E.g. $f(x, y) = x^2 + y$ has functions $f_x(y) = x^2 + y$ and $f_y(x) = x^2 + y$
- **Partial derivative** w.r.t. one variable keeps other variables constant

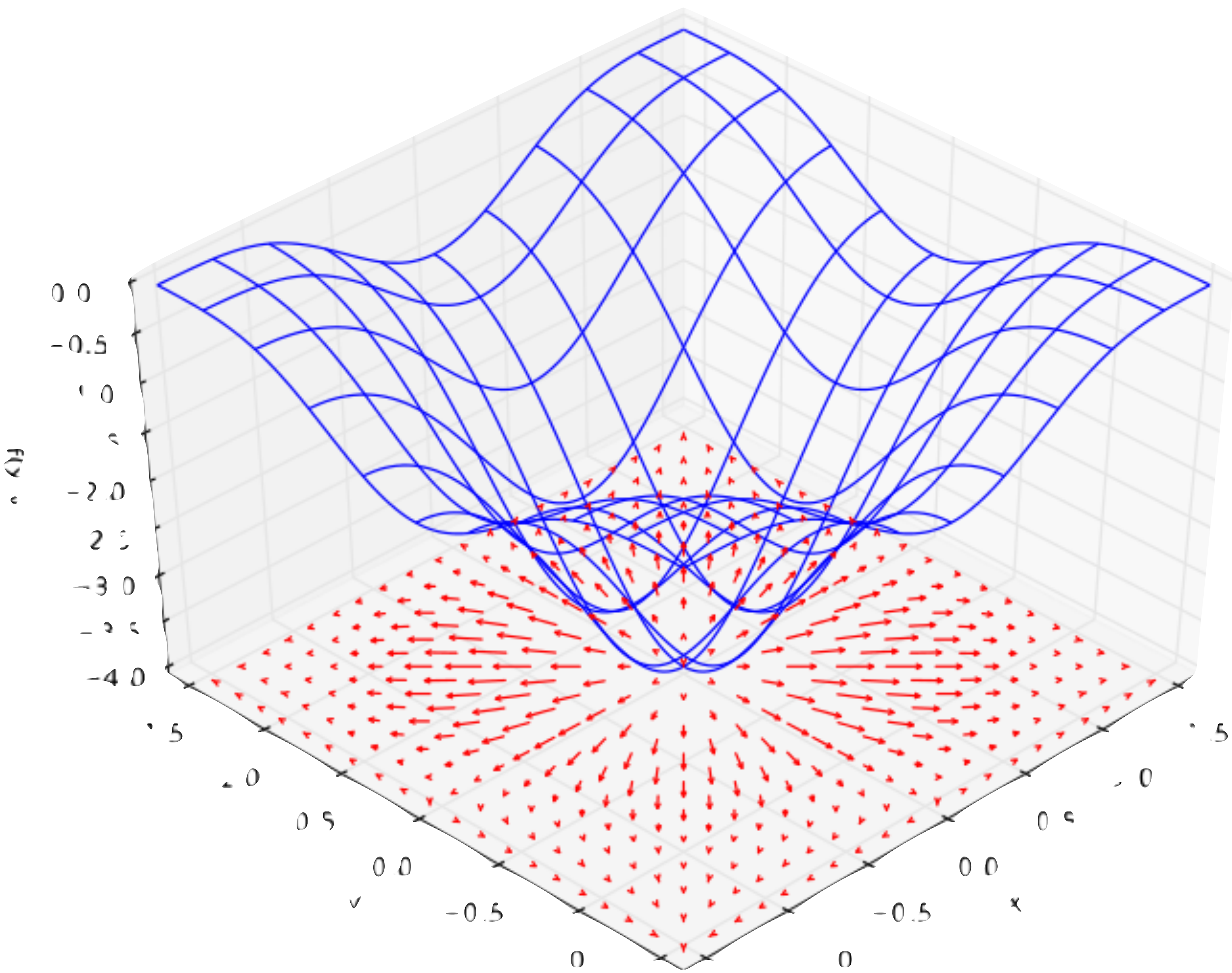
$$\frac{\partial f}{\partial x}(x, y) = f_y'(x) = 2x$$

Gradient



- **Gradient** is the derivative for multivariate functions $f\colon \mathbb{R}^n \to \mathbb{R}$
  - $\nabla f = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_n}\right)$
  - Here (and later), we assume that the derivatives exist
- Gradient is a function $\nabla f\colon \mathbb{R}^n \to \mathbb{R}^n$
- $\nabla f(\boldsymbol{x})$ points “up” in the function at point $\boldsymbol{x}$

Gradient



Hessian

- **Hessian** is a square matrix of all second-order partial derivatives of a function $f\colon \mathbb{R}^n \to \mathbb{R}$

$$\boldsymbol{H}(f) = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{pmatrix}$$

- As usual, we assume the derivatives exist

Jacobian matrix

- If $f\colon \mathbb{R}^m \to \mathbb{R}^n$, then its **Jacobian** (matrix) is an $n{\times}m$ matrix of partial derivatives in form

$$\boldsymbol{J} = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_m} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & & \frac{\partial f_2}{\partial x_m} \\ \vdots & & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_m} \end{pmatrix}$$

- Jacobian is the best linear approximation of f
- $\boldsymbol{H}(f(\boldsymbol{x})) = \boldsymbol{J}(\nabla f(\boldsymbol{x}))^T$

Examples

Function

$$f(x,y) = x^2 + 2xy + y$$

Partial derivatives

$$\frac{\partial f}{\partial x}(x,y) = 2x + 2y$$
$$\frac{\partial f}{\partial y}(x,y) = 2x + 1$$

Gradient

$$\nabla f = (2x + 2y, 2x + 1)$$

Hessian

$$\boldsymbol{H}(f) = \begin{pmatrix} 2 & 2 \\ 2 & 0 \end{pmatrix}$$

Function

$$f(x,y) = \begin{pmatrix} x^2 y \\ 5x + \sin y \end{pmatrix}$$

Jacobian

$$\boldsymbol{J}(f) = \begin{pmatrix} 2xy & x^2 \\ 5 & \cos y \end{pmatrix}$$

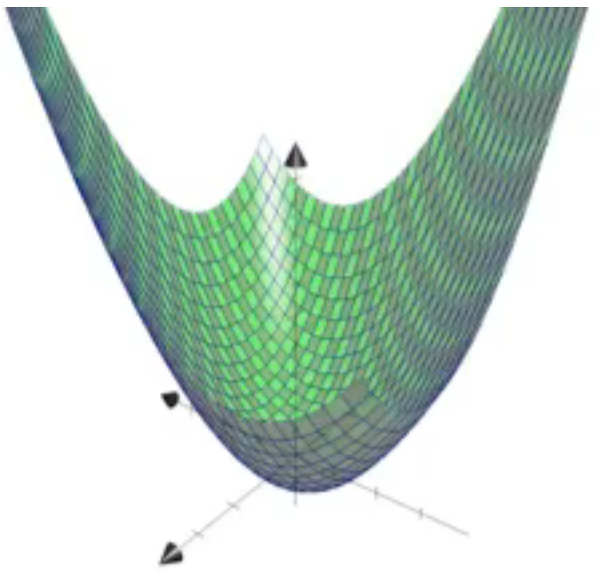
Gradient's properties

- **Linearity:** $\nabla(\alpha f + \beta g)(\boldsymbol{x}) + \alpha\nabla f(\boldsymbol{x}) + \beta\nabla g(\boldsymbol{x})$
- **Product rule:** $\nabla(fg)(\boldsymbol{x}) = f(\boldsymbol{x})\nabla g(\boldsymbol{x}) + g(\boldsymbol{x})\nabla f(\boldsymbol{x})$
- **Chain rule:**

  IMPORTANT!

  - If $f\colon \mathbb{R}^n \to \mathbb{R}$ and $g\colon \mathbb{R}^m \to \mathbb{R}^n$, then
    $\nabla(f \circ g)(\boldsymbol{x}) = \boldsymbol{J}(g(\boldsymbol{x}))^T(\nabla f(\boldsymbol{y}))$ where $\boldsymbol{y} = g(\boldsymbol{x})$
  - If f is as above and $h\colon \mathbb{R} \to \mathbb{R}$, then
    $\nabla(h \circ f)(\boldsymbol{x}) = h'(f(\boldsymbol{x}))\nabla f(\boldsymbol{x})$

Convexity



- A function is **convex** if any line segment between two points of the function lie above or on the graph
- For univariate f, if $f''(x) \geq 0$ for all x
- For multivariate f, if its Hessian is **positive semidefinite**
  - I.e. $\boldsymbol{z}^T\boldsymbol{H}\boldsymbol{z} \geq 0$ for any $\boldsymbol{z}$
- Convex function's local minimum is its global minimum

Preserving the convexity

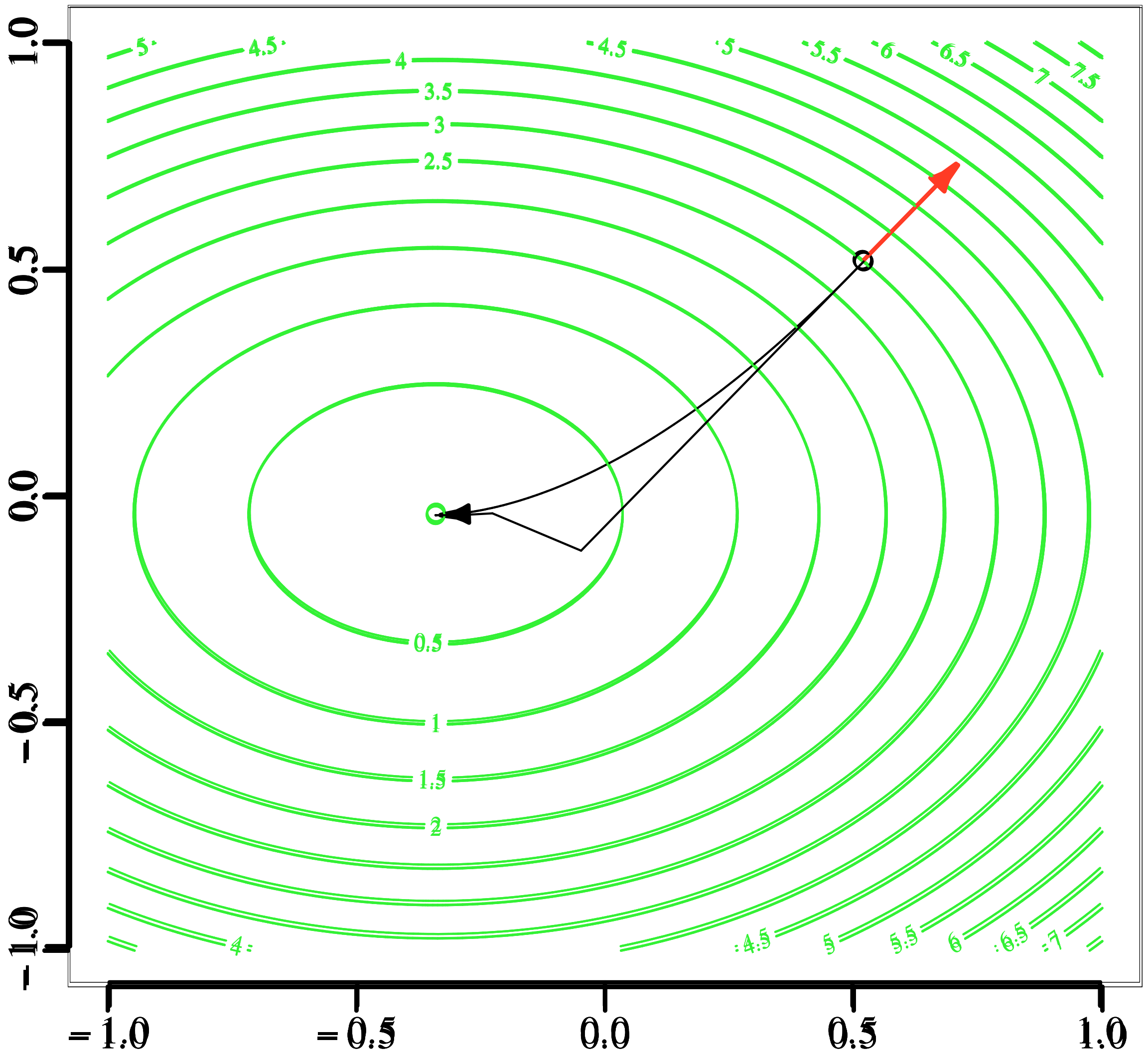
- If f is convex and $\lambda > 0$, then λf is convex
- If f and g are convex, the $f + g$ is convex
- If f is convex and g is **affine** (i.e. $g(\boldsymbol{x}) = \boldsymbol{Ax} + \boldsymbol{b}$), then $f \circ g$ is convex (N.B. $(f \circ g)(\boldsymbol{x}) = f(\boldsymbol{Ax} + \boldsymbol{b})$)
- Let $f(\boldsymbol{x}) = (h \circ g)(\boldsymbol{x})$ with $g: \mathbb{R}^n \to \mathbb{R}$ and $h: \mathbb{R} \to \mathbb{R}$; f is convex if
  - g is convex and h is nondecreasing and convex
  - g is concave and h is non-increasing and convex

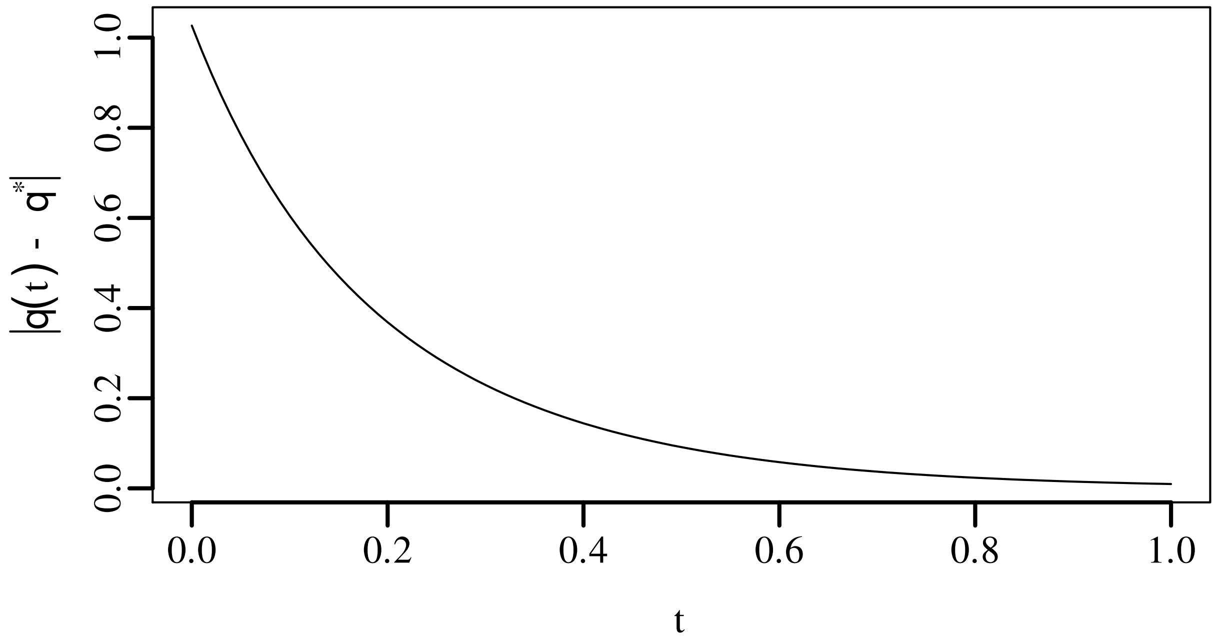
Gradient descent

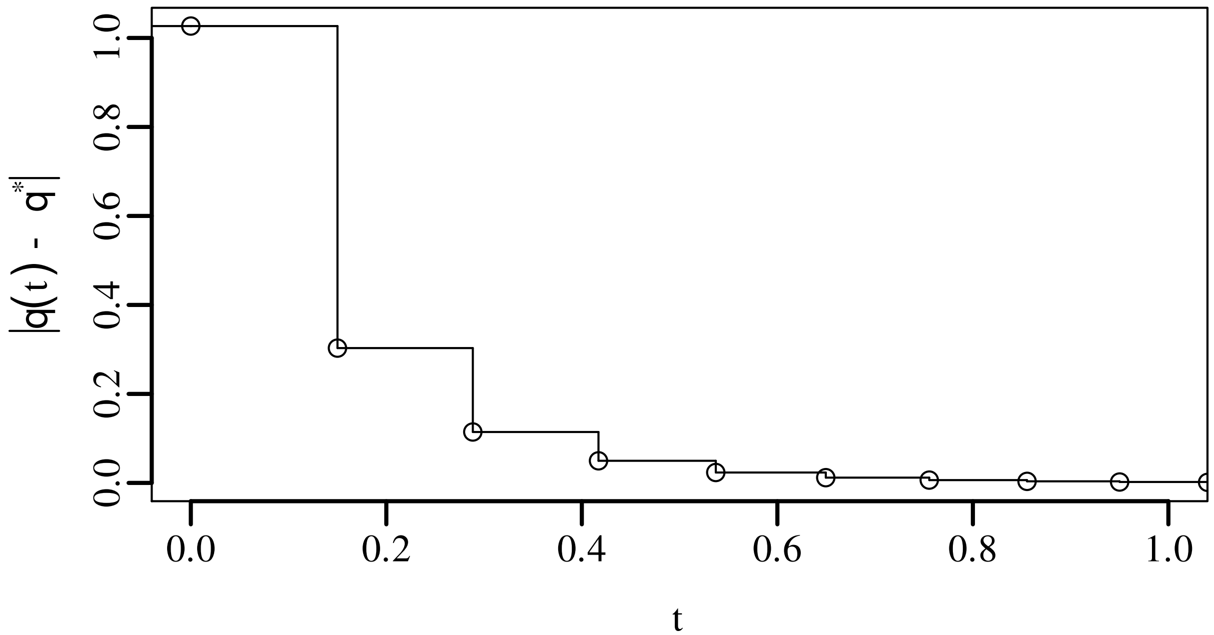
Idea

- If f is convex, we should find it's minimum by following its negative gradient
  - But the gradient at $\boldsymbol{x}$ points to minimum only at $\boldsymbol{x}$
  - Hence, we need to descent slowly down the gradient

Example







Gradient descent

- Start from random point $\boldsymbol{x}^0$
- At step n, update $\boldsymbol{x}^n \leftarrow \boldsymbol{x}^{n-1} - \gamma \nabla f(\boldsymbol{x}^{n-1})$
  - γ is some small **step size**
  - Often, γ depends on the iteration
    $\boldsymbol{x}^n \leftarrow \boldsymbol{x}^{n-1} - \gamma_n \nabla f(\boldsymbol{x}^{n-1})$
- With suitable f and step size, will converge to local minimum

Example: least squares

- Given $\boldsymbol{A} \in \mathbb{R}^{n \times m}$ and $\boldsymbol{b} \in \mathbb{R}^{n}$, find $\boldsymbol{x} \in \mathbb{R}^{m}$ s.t. $\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|^2/2$ is minimized
  - Can be solved using SVD…
- Calculate the gradient of $f_{\boldsymbol{A},\boldsymbol{b}}(\boldsymbol{x}) = \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|^2/2$
- Employ the gradient descent approach
  - In this case, the step size can be calculated analytically

Example: the gradient

Let's write open:

$$
\begin{aligned}
\frac{1}{2}\|\boldsymbol{Ax}-\boldsymbol{b}\|^2 &= \frac{1}{2}\sum_{i=1}^{n}\big((\boldsymbol{Ax})_i - b_i\big)^2 \\
&= \frac{1}{2}\sum_{i=1}^{n}\Big(\sum_{j=1}^{m} a_{ij}x_j - b_i\Big)^2 \\
&= \frac{1}{2}\sum_{i=1}^{n}\Big(\Big(\sum_{j=1}^{m} a_{ij}x_j\Big)^2 - 2b_i\sum_{j=1}^{m} a_{ij}x_j + b_i^2\Big) \\
&= \frac{1}{2}\sum_{i=1}^{n}\Big(\sum_{j=1}^{m} a_{ij}x_j\Big)^2 - \sum_{i=1}^{n} b_i\sum_{j=1}^{m} a_{ij}x_j + \frac{1}{2}\sum_{i=1}^{n} b_i^2
\end{aligned}
$$

Example: the gradient

The partial derivative w.r.t. x_j:

$$\frac{\partial}{\partial x_j}\Big(\frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|^2\Big) = \frac{\partial}{\partial x_j}\Big(\frac{1}{2}\sum_{i=1}^{n}\Big(\sum_{k=1}^{m} a_{ik}x_k\Big)^2 - \sum_{i=1}^{n} b_i \sum_{k=1}^{m} a_{ik}x_k + \frac{1}{2}\sum_{i=1}^{n} b_i^2\Big)$$

Linearity

$$= \frac{1}{2}\sum_{i=1}^{n}\frac{\partial}{\partial x_j}\Big(\sum_{k=1}^{m} a_{ik}x_k\Big)^2 - \sum_{i=1}^{n} b_i \frac{\partial}{\partial x_j}\sum_{k=1}^{m} a_{ik}x_k + \frac{\partial}{\partial x_j}\frac{1}{2}\sum_{i=1}^{n} b_i^2$$

= 0 if k ≠ j

= 0

$$= \frac{1}{2}\sum_{i=1}^{n}\frac{\partial}{\partial x_j}\Big(\sum_{k=1}^{m} a_{ik}x_k\Big)^2 - \sum_{i=1}^{n} b_i a_{ij}$$

Chain rule

$$= \sum_{i=1}^{n} a_{ij}\sum_{k=1}^{m} a_{ik}x_k - \sum_{i=1}^{n} b_i a_{ij}$$

$$= \sum_{i=1}^{n} a_{ij}\Big(\sum_{k=1}^{m} a_{ik}x_k - b_i\Big)$$

Example: the gradient

Collecting terms:

Matrix product

$$\frac{\partial}{\partial x_j}\left(\frac{1}{2}\|\boldsymbol{Ax}-\boldsymbol{b}\|^2\right) = \sum_{i=1}^{n} a_{ij}\left(\sum_{k=1}^{m} a_{ik}x_k - b_i\right)$$

$$= \sum_{i=1}^{n} a_{ij}\left((\boldsymbol{Ax})_i - b_i\right)$$

Another matrix product

$$= \left(\boldsymbol{A}^T(\boldsymbol{Ax}-\boldsymbol{b})\right)_j$$

Hence we have:

$$\nabla\left(\frac{1}{2}\|\boldsymbol{Ax}-\boldsymbol{b}\|^2\right) = \boldsymbol{A}^T(\boldsymbol{Ax}-\boldsymbol{b})$$

Example: the gradient

The other way: Use the chain rule

$$\nabla\Big(\frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|^2\Big) = \boldsymbol{J}(\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b})^T\Big(\nabla(\frac{1}{2}\|\boldsymbol{y}\|^2)\Big) \quad \boldsymbol{y}=\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}$$

$$= \boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b})$$

Gradient descent & matrices

- How about "Given $\boldsymbol{A}$, find small $\boldsymbol{B}$ and $\boldsymbol{C}$ s.t. $||\boldsymbol{A} - \boldsymbol{BC}||_F$ is minimized"?
  - Not convex for $\boldsymbol{B}$ and $\boldsymbol{C}$ jointly
- Fix some $\boldsymbol{B}$ and solve for $\boldsymbol{C}$
  - $\boldsymbol{C} = \text{argmin}_{\boldsymbol{X}} ||\boldsymbol{A} - \boldsymbol{BX}||_F$
- Use the found $\boldsymbol{C}$ and solve for $\boldsymbol{B}$, and repeat until convergence

How to solve for $\boldsymbol{C}$?

- $\boldsymbol{C} = \text{argmin}_{\boldsymbol{X}} \|\boldsymbol{A} - \boldsymbol{B}\boldsymbol{X}\|_F$ still needs some work
- Write the norm as sum of column-wise errors

  $\|\boldsymbol{A} - \boldsymbol{B}\boldsymbol{X}\|_F = \sum \|\boldsymbol{a}_j - \boldsymbol{B}\boldsymbol{x}_j\|_2$

  - Now the problem is a series of standard least-squares problems
  - Each can be solved independently

How to select the step size?

- Recall: $\boldsymbol{x}^n \leftarrow \boldsymbol{x}^{n-1} - \gamma_n \nabla f(\boldsymbol{x}^{n-1})$
- Selecting correct γ_n for each n is crucial
  - Methods for optimal step size are often slow (e.g. line search)
  - Wrong step size can lead to non-convergence

Stochastic gradient descent

Basic idea

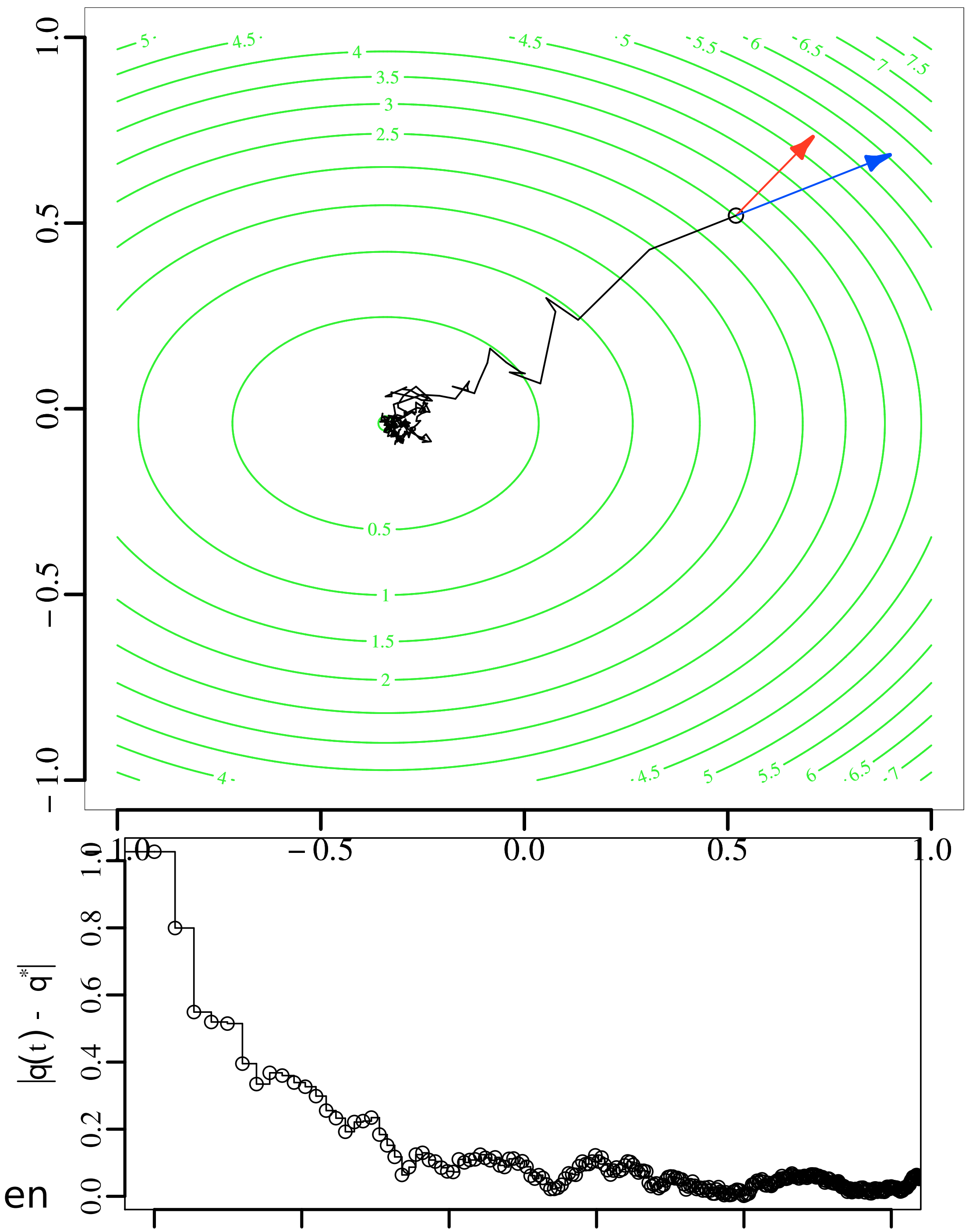
- With gradient descent, we need to calculate the gradient for $\boldsymbol{c} \mapsto ||\boldsymbol{a} - \boldsymbol{Bc}||$ many times for different $\boldsymbol{a}$ in each iteration
- Instead we can fix one element a_{ij} and update the ith row of $\boldsymbol{B}$ and jth column of $\boldsymbol{C}$ accordingly
- When we choose a_{ij} randomly, this is **stochastic gradient descent** (SGD)

Local gradient

- With fixed a_{ij}, $||a_{ij} - (\boldsymbol{BC})_{ij}|| = a_{ij} - \sum b_{ik} c_{kj}$
  - Local gradient for b_{ik} is $-2c_{kj}(a_{ij} - (\boldsymbol{BC})_{ij})$
  - Similarly for c_{kj}
- This allows us to update the factors by only computing one gradient
  - Gradient needs to be sufficiently scaled

SGD process

- Initialize with random ***B*** and ***C***
- **repeat**
  - Pick a random element (i, j)
  - Update a row of ***B*** and a column of ***C*** using the local gradients w.r.t. a_{ij}



SGD pros and cons

- Each iteration is faster to compute
  - But can increase the error
- Does not need to know all elements of the input data
  - Scalability
  - Partially observed matrices (e.g. collaborative filtering)
- The step size still needs to be chosen carefully

Newton’s method

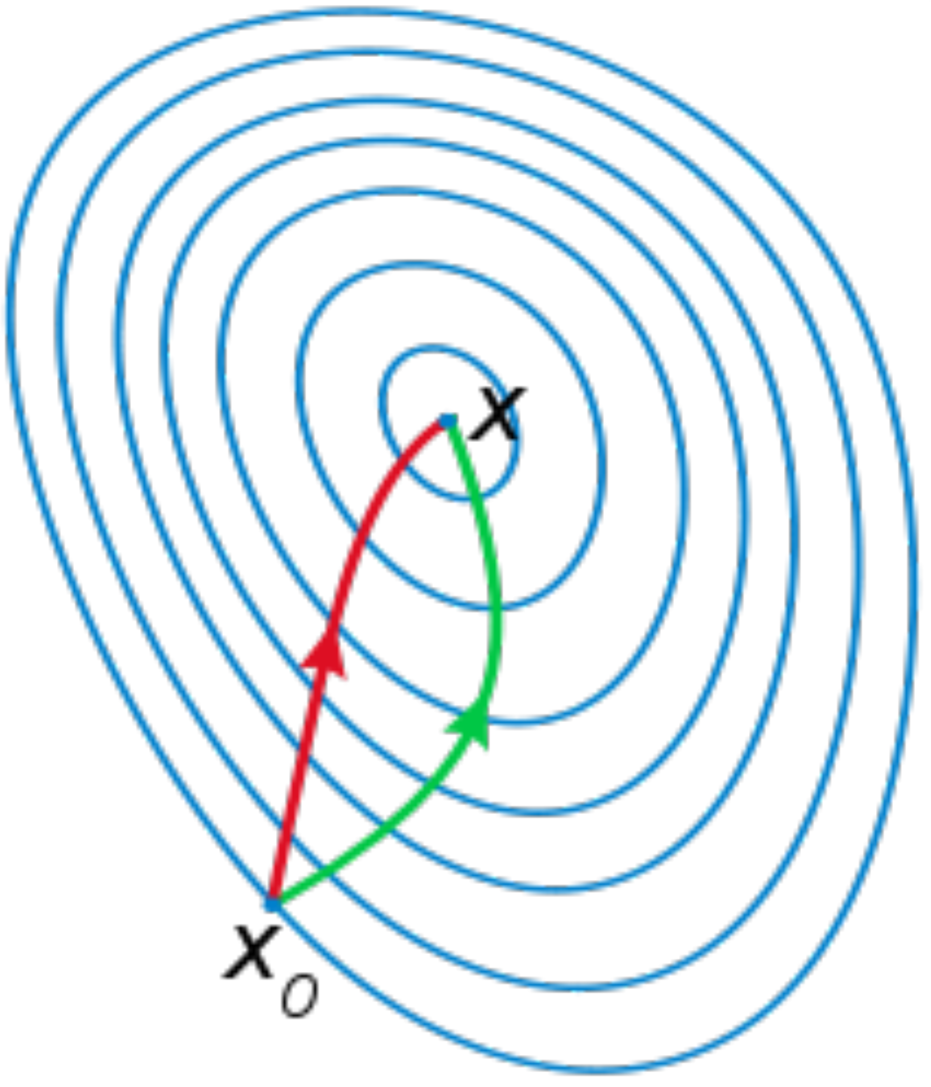
Basic idea

- Iterative update rule:

  $\boldsymbol{x}_{n+1} \leftarrow \boldsymbol{x}_n - [\boldsymbol{H}(f(\boldsymbol{x}_n))]^{-1} \nabla f(\boldsymbol{x}_n)$

  - Assuming Hessian exists and is invertible…

- Takes curvature information into account



Pros and cons

- Much faster convergence
  - But Hessian is slow to compute and takes lots of memory
- Quasi-Newton methods (e.g. L-BFGS) compute the Hessian indirectly
- Often still needs some step size other than 1

Alternating least squares

Basic idea

- Given $\boldsymbol{A}$ and $\boldsymbol{B}$, we can find $\boldsymbol{C}$ that minimizes $||\boldsymbol{A} - \boldsymbol{B}\boldsymbol{C}||_F$
  - In gradient descent, we move slightly towards $\boldsymbol{C}$
- In **alternating least squares** (ALS), we replace $\boldsymbol{C}$ with the new one

Basic ALS algorithm

- Given $\boldsymbol{A}$, sample a random $\boldsymbol{B}$
- **repeat** until convergence
  - $\boldsymbol{C} \leftarrow \operatorname{argmin}_{\boldsymbol{X}} \|\boldsymbol{A} - \boldsymbol{B}\boldsymbol{X}\|_F$
  - $\boldsymbol{B} \leftarrow \operatorname{argmin}_{\boldsymbol{X}} \|\boldsymbol{A} - \boldsymbol{X}\boldsymbol{C}\|_F$

ALS pros and cons

- Can have faster convergence than gradient descent (or SGD)
- The update is slower to compute than in SGD
  - About as fast as in gradient descent
- Requires fully-observed matrices

Adding constraints

The problem setting

- So far, we have done **unconstrained optimization**
- What if we have constrains on the optimal solution?
  - E.g. all matrices must be nonnegative
- In general, the above approaches won’t admit these constraints

General case

- Minimize $f(\boldsymbol{x})$
- Subject to

  $g_i(\boldsymbol{x}) \leq 0, i = 1, \ldots, m$

  $h_j(\boldsymbol{x}) = 0, j = 1, \ldots, k$
- Assuming certain regularity conditions, there exists constraints μ_i (i=1,…,m) and λ_j (j=1,…,k) that satisfy **Karush–Kuhn–Tucker** (KKT) conditions

KKT conditions

- Let $\boldsymbol{x}^*$ be the optimal solution
- **Stationarity:**
  - $-\nabla f(\boldsymbol{x}^*) = \sum_i \mu_i \nabla g_i(\boldsymbol{x}^*) + \sum_j \lambda_j \nabla h_j(\boldsymbol{x}^*)$
- **Primal feasibility:**
  - $g_i(\boldsymbol{x}^*) \leq 0$ for all $i = 1, \ldots, m$
  - $h_j(\boldsymbol{x}^*) = 0$ for all $j = 1, \ldots, k$
- **Dual feasibility:**
  - $\mu_i \geq 0$ for all $i = 1, \ldots, m$
- **Complementary slackness:**
  - $\mu_i g_i(\boldsymbol{x}^*) = 0$ for all $i = 1, \ldots, m$

When do KKT conditions hold

- KKT conditions hold under certain regularity conditions
  - E.g. g_i and h_j are affine
  - Or f is convex and exists $\boldsymbol{x}$ s.t. $h(\boldsymbol{x}) = 0$ and $g_i(\boldsymbol{x}) < 0$
- Nonnegativity is an example of linear (hence, affine) constraint

What to do with the KKT conditions?

- $\boldsymbol{\mu}$ and $\boldsymbol{\lambda}$ are new unknown variables
    - Must be optimized together with $\boldsymbol{x}$
- The conditions appear in the optimization
    - E.g. in the gradient
- The KKT conditions are rarely solved directly

Summary

- There are **many** methods for optimization
  - We only scratched the surface
- Methods are often based on gradients
  - Can lead into ugly equations
- Next week: applying these techniques for finding nonnegative factorizations… *Stay tuned!*